A DECODER FOR A WIRELESS COMMUNICATION DEVICE

The present invention relates to a decoder for a wireless communication device.

Wireless communication systems are widely deployed to provide various types of communications such as voice and data. One such system is wideband code division multiple access WCDMA, which has been adopted in various competing wireless communication standards, for example 3rd generation partnership project 3GPP and 3GPP2.

To overcome data corruption that can occur during RF transmission the different wireless communication standards typically include some form of channel coding, where one common channel coding technique is turbo coding.

Turbo coding involves the use of a turbo encoder for encoding a code segment (i.e. a data packet) and a turbo decoder for the decoding of the encoded code segment.

A turbo encoder includes two convolutional encoders and an interleaver, where the interleaver shuffles (i.e. interleaves) the information bits in the packet in accordance with a specified interleaving scheme.

The turbo encoder uses a first convolutional encoder to encode information bits within a packet to generate a first sequence of parity bits in parallel to the interleaver shuffling the information bits, where the shuffled information bits are encoded by a second encoder to generate a second sequence of parity bits. The information bits and the parity bits in the first and second sequence are then modulated and transmitted to a receiver.

The information bits and the first and second sequence of parity bits are received by a receiver and decoded by a turbo decoder.

The turbo decoder initially stores the received information bits and the parity bits in the first and second sequence in a buffer. Initially, the information bits and the first sequence of parity bits from the first convolutional encoder are retrieved from the buffer and decoded by a first soft input soft output SISO decoder to provide 'extrinsic' information indicative of adjustments in the confidence in the detected values for the information bits. Intermediate results that include the extrinsic information from the first SISO decoder are then stored in the buffer in an interleaved order matching the code interleaving used at the transmitter.

The intermediate results, the information bits and the second sequence of parity bits from the second encoder are retrieved from the buffer and decoded by a second SISO decoder to provide extrinsic information indicative of further adjustments in the confidence in the detected values for the information bits. Intermediate results that comprise the extrinsic information from the second SISO decoder are then stored in the buffer in a deinterleaved order complementary to the code interleaving performed at the transmitter. The intermediate results are used in a next decoding iteration performed by the turbo decoder. The turbo decoder performs a predetermined number of decoding iterations before producing a decision on the value of the decoded information bit.

Commonly used algorithms used within SISO decoders are the maximum a posteriori MAP decoding algorithm and the log MAP decoding algorithm. The log MAP decoding algorithm is analogues to the MAP decoding algorithm but performed in the logarithmic domain.

The MAP decoding algorithm uses forward state metrics, commonly referred to as alphas α, and backward state metrics, commonly referred to as betas β, to

determine soft output results, where the forward state metrics α and backward state metrics β characterise a state in a trellis structure.

The MAX* function is used within the log-MAP algorithm and is represented by MAX*(a(n),b(n)), where a(n) and b(n) are inputs to the MAX* function. The inputs a(n) and b(n) can be forward state metrics, backward state metrics or a combination of both.

The MAX*(a(n),b(n)) function is equal to MAX(a(n),b(n)) plus a correction value where the correction value is equal to log(1+exp(-$|a(n)-b(n)|$)).

The MAX(a(n),b(n)) term of the equation is usually straight forward to calculate, however the correction value is relatively complicated to calculate and is usually approximated using either a linear approximation, a step approximation or a look-up table.

As the state metric calculations are performed within the SISO decoder the values within the accumulated path metrics can overflow leading to incorrect results.

One solution to the overflow problem involves the use of modulo arithmetic's. A modulo n operation on a number provides the remainder when the number is divided by n, for example 10(binary 1010)modulo 8 = 2(binary 010) and 28(binary 11100)modulo 16 = 12(binary 1100). Consequently, as can be seen from the examples, determine a value for a modulo operation where the remainder is a value to the power of two is simply a question of masking off any unwanted bits.

The modulo function, as illustrated in figure 1, can be regarded as a sawtooth function.

An alternative implementation of the modulo function can be defined by: xmodF = $x - 2F\left\lfloor \left(\frac{x+F}{2F} \right) \right\rfloor$, which allows negative numbers to be accommodated. This function is illustrated in figure 2.

It is desirable to have an apparatus and method for generating a linearly approximated MAX* log MAP algorithm that operates on modulo functions.

In accordance with a first aspect of the present invention there is provided a decoder for a wireless communication device according to claim 1.

In accordance with a second aspect of the present invention there is provided a method for generating a MAX* value according to claim 8.

An embodiment of the invention will now be described, by way of example, with reference to the drawings, of which:

Figure 1 illustrates a graphical representation of a first modulo function;

Figure 2 illustrates a graphical representation of a second modulo function;

Figure 3 illustrates a graphical representation of the variation in the MAX* correction term versus $|a(n) - b(n)|$;

Figure 4 illustrates a decoder according to an embodiment of the present invention.

The curve A in figure 3 illustrates the correction term for the MAX*(a(n),b(n)) function (i.e. MAX*(a(n),b(n)-MAX(a(n),b(n)) as a function of $|a(n)-b(n)|$), where $|a(n)-b(n)|$ is the absolute value of the difference between a(n) and b(n)).

As can be seen from curve A the correction term is greatest for low values of $|a(n)-b(n)|$ and gradually decreases to zero as $|a(n)-b(n)|$ increases.

As stated above, an easy technique for approximating the correction term is the use of linear approximation, as illustrated by line B in figure 3. As illustrated, the linear approximation provides a close approximation for the correction term for low values of $|a(n)-b(n)|$. The intersection of the line B on the $|a(n)-b(n)|$ axis indicates the $|a(n)-b(n)|$ value above which the linear approximation correction term goes to zero. Consequently, using linear approximation, the intersection point determines a threshold value, designated C, for determining if a correction value is to be applied to $|a(n)-b(n)|$, where the intersection point is defined by the linear approximation equation.

The use of the linear approximation technique allows easy calculation of the MAX* function, as described below.

One suitable linear approximation equation (i.e. the correction term used) is given by MAX(0, $(C-|a(n)-b(n)|)/2$).

Consequently, the MAX* function can be written as:

MAX(a(n),b(n)) + MAX(0, $(C-|a(n)-b(n)|)/2$)

=MAX(0+MAX(a(n),b(n)), $(C-|a(n)-b(n)|)/2$ +MAX(a(n),b(n))

=MAX(a(n),b(n), $(C-|a(n)-b(n)|)/2$ +MAX(a(n),b(n))

=MAX(a(n),b(n), $(a(n)+b(n)+C)/2$)

To minimise the problem of accumulated state metric overflow, as discussed above, the above terms are converted into their corresponding 'modF' values where F is selected such that $|a(n)-b(n)|$ <F. F is chosen by analysing the algorithm and determining what would be the maximum possible value of $|a(n)-b(n)|$ for any a(n) and b(n) that can enter the MAX* function.

To ease the hardware implementation for handling the modulo value F is preferable a value to the power of two.

The modF of a(n) becomes a(n)modF.

The modF of b(n) becomes b(n)modF.

However, the modF of (a(n)+b(n)+C)/2 is

$$\left(a(n) \bmod F + \frac{(b(n) \bmod F - a(n) \bmod F) \bmod F + C}{2}\right) \bmod F$$

and not $\left(\frac{a(n) \bmod F + b(n) \bmod F + C}{2}\right) \bmod F$.

This is demonstrated by the following:

The equation xmodF is equivalent to $x - 2F\left\lfloor \left(\frac{x+F}{2F} \right) \right\rfloor$, where the $\lfloor x \rfloor$ term is the floor of 'x'.

Accordingly:

(a(n)+C)modF = (a(n)modF+C)modF

and

(a(n)-b(n))modF = a(n) – b(n) if and only if $|a(n) - b(n)|$ <F

Using these two identities proves:

$$\left(\frac{a(n) + b(n) + C}{2} \right) \bmod F = \left(a(n) + \left(\frac{b(n) - a(n) + C}{2} \right) \right) \bmod F$$

$$= \left(a(n) \bmod F + \left(\frac{(b(n) - a(n)) \bmod F + C}{2} \right) \right) \bmod F$$

$$= \left(a(n) \bmod F + \frac{((b(n) \bmod F - a(n) \bmod F) \bmod F + C)}{2} \right) \bmod F .$$

For values of C<F/2 an alternative implementation of the modulo of a linear approximation of a MAX function is equal to:

$$\left(\left(\frac{(a(n) \bmod F + C) \bmod F + b(n) \bmod F}{2}\right) \bmod F + F * s\right) \bmod F$$

where s is calculated from the binary expression s=[a(m) xor b(m)] and [((a(m) xor a(m-1)) and ((b(m) xor b(m-1)], where a and b are represented by m bits so that a(m) is the most significant bit of a and a(m-1) is next to the most significant bit.

This algorithm is easy to calculate in silicon as s involves only binary operations and F is chosen to be a power of two.

A decoder 400 for implementing the above MAX* equation is shown in figure 4 and is arranged to output MAX(a(n)modF,b(n)modF) when $|a(n) - b(n)|$ is greater than the threshold value C and to output $\left(a(n) \bmod F + \frac{((b(n) \bmod F - a(n) \bmod F) \bmod F + C)}{2}\right) \bmod F$ when $|a(n) - b(n)|$ is less than the threshold value C. If $|a(n) - b(n)|$ equals C then either MAX(a(n)modF,b(n)modF) or $\left(a(n) \bmod F + \frac{((b(n) \bmod F - a(n) \bmod F) \bmod F + C)}{2}\right) \bmod F$ can be output.

The decoder 400 includes a first substracting unit 401, a second substracting unit 402, a calculator 403 in the form of an adder unit and a selector 404 in the form of a multiplexer unit. The first substracting unit 401, the second substracting unit 402 and the adder unit 403 are each arranged to receive a(n)modF, b(n)modF and the threshold value C.

The first substracting unit 401 is arranged to generate the sign of (b(n)modF-a(n)modF-C)modF. The second substracting unit 402 is arranged to generate the sign of (a(n)modF-b(n)modF-C)modF. The adder unit 403 is arranged to generate $\left(a(n)\bmod F+\frac{((b(n)\bmod F-a(n)\bmod F)\bmod F+C)}{2}\right)\bmod F$, where the division by two corresponds to a shift in bit position by one.

The modF operation is performed by ignoring the overflow (i.e. the carry bit of the msb bit addition is ignored).

The output from the first substracting unit 401, the second substracting unit 402 and the adding unit 403 (i.e. the sign of (b(n)modF-a(n)modF-C)modF, the sign of (a(n)modF-b(n)modF-C)modF and $\left(a(n)\bmod F+\frac{((b(n)\bmod F-a(n)\bmod F)\bmod F+C)}{2}\right)\bmod F$ respectively) are provided to the multiplex unit 404 along with the values of a(n)modF and b(n)modF.

The multiplexer 404 is arranged to output a MAX*(a(n)modF, b(n)modF) equal to a(n)modF when the sign of (a(n)modF-b(n)modF-C)modF is positive and the sign of (b(n)modF-a(n)modF-C)modF is negative.

The multiplexer 404 is arranged to output a MAX*(a(n)modF, b(n)modF) equal to b(n)modF when the sign of (a(n)modF-b(n)modF-C)modF is negative and the sign of (b(n)modF-a(n)modF-C)modF is positive.

The multiplexer 404 is arranged to output a MAX*(a(n)modF, b(n)modF) equal to $\left(a(n)\bmod F+\frac{((b(n)\bmod F-a(n)\bmod F)\bmod F+C)}{2}\right)\bmod F$ when the sign of (a(n)modF-b(n)modF-C)modF is negative and the sign of (b(n)modF-a(n)modF-C)modF is negative.

It will be apparent to those skilled in the art that the disclosed subject matter may be modified in numerous ways and may assume many embodiments other than the preferred forms specifically set out as described above, for example the above embodiments could be arranged such that the modulo for other linear approximation equations can be calculated and an additional substracting unit could be used to determine the sign of a(n)-b(n) to assist the selection process.